Digital Image Processing & Machine Vision

Lecture # 5: Spatial Filtering

Spatial Filtering

- Some neighborhood operations work with the values of the image pixels in the neighborhood and the corresponding values of a sub-image that has the same dimensions as the neighborhood.
- The sub-image is called a filter, mask, kernel, template, or window, with the first three terms being the most prevalent terminology.
- The values in a filter sub-image are referred to as coefficients, rather than pixels.
- The term spatial filtering is used to differentiate from the more traditional frequency domain filtering

Spatial Filtering

- For linear spatial filtering, the response is given by a sum of products of the filter coefficients and the corresponding image pixels in the area spanned by the filter mask

$$R = w(-1,-1)f(x-1,y-1) + w(-1,0)f(x-1,y) + \cdots + w(0,0)f(x,y) + \cdots + w(1,0)f(x+1,y) + w(1,1)f(x+1,y+1),$$

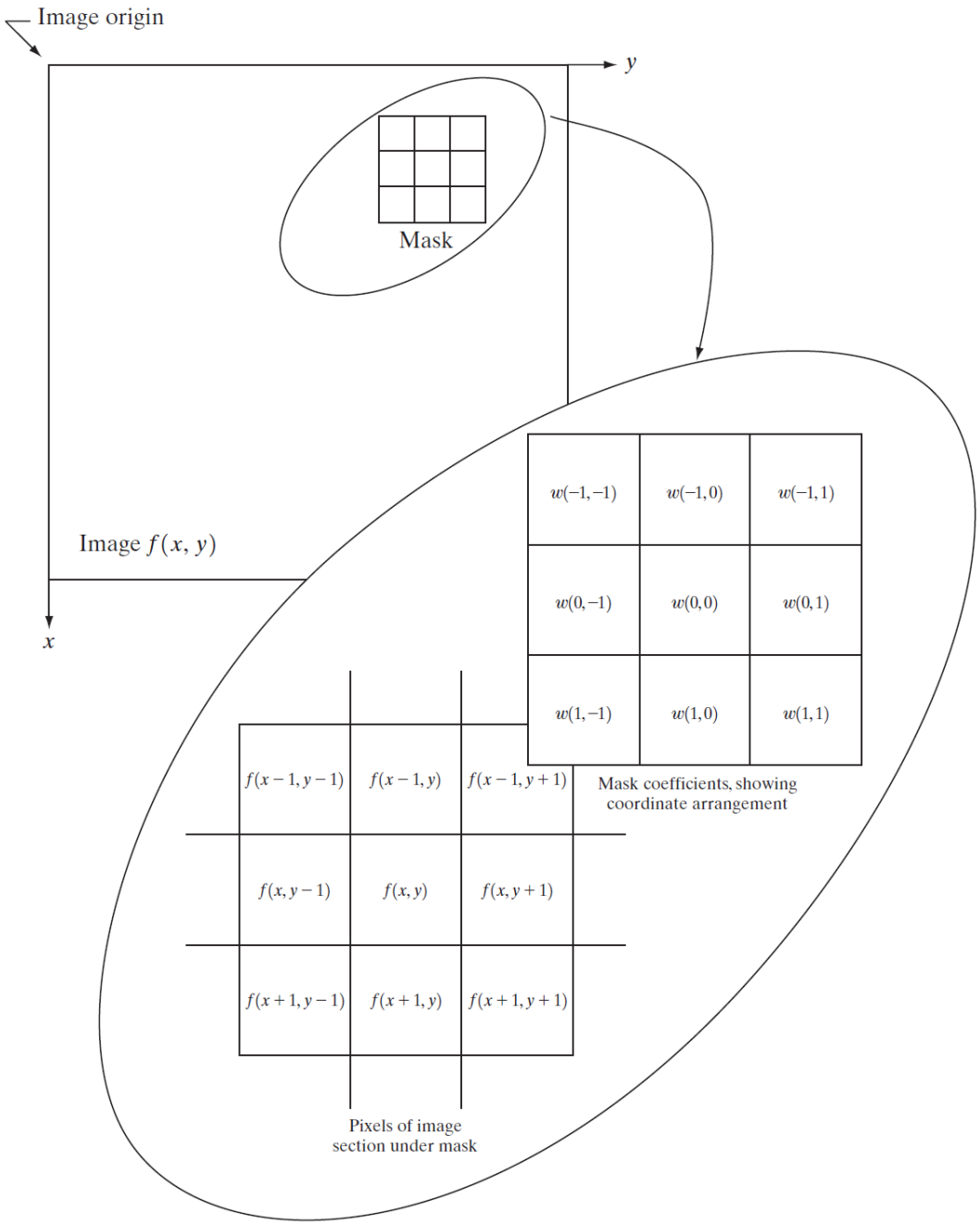
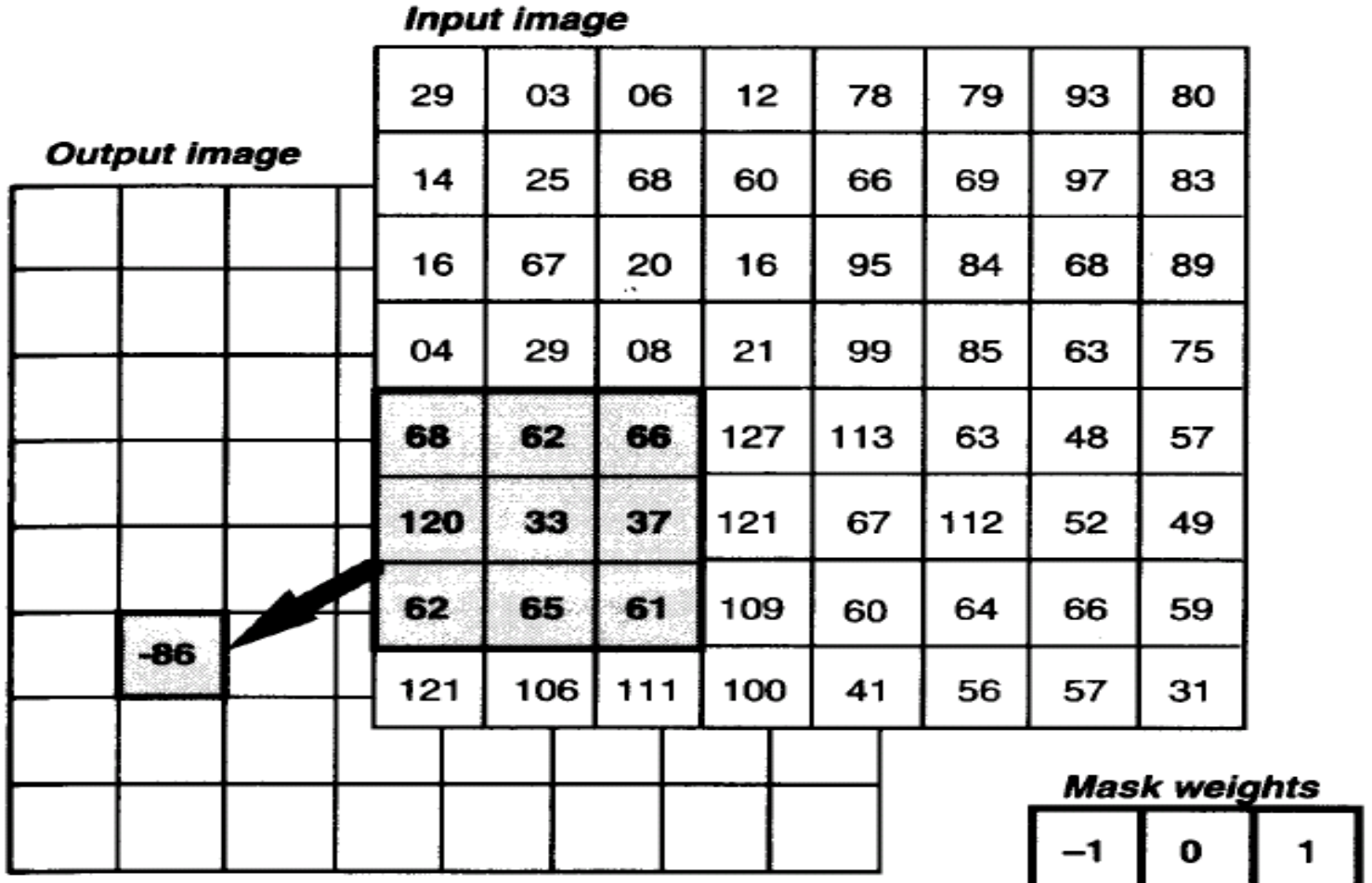


FIGURE 3.32 The mechanics of spatial filtering. The magnified drawing shows a 3×3 mask and the image section directly under it; the image section is shown displaced out from under the mask for ease of readability.



Mask weights

–1	0	1
–1	0	1
–1	0	1

Output value is -86, because:

(–1x68) + (0x62) + (1x66) + (–1x120) + (0x33)
+ (1x37) + (–1x62) + (0x65) + (1x61)
= –86

Spatial Filtering

w(-1,-1)	w(-1,0)	w(-1,1)
w(0,-1)	w(0,0)	w(0,1)
w(1,-1)	w(1,0)	w(1,1)

Mask coefficients showing coordinate arrangement

Convolution

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x-s, y-t)$$

$$g = \omega * f$$

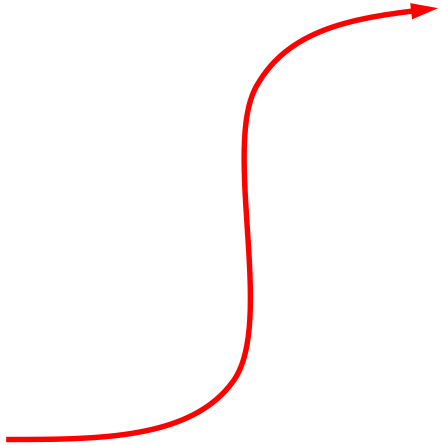
Convolution

Convolution kernel, ω

1	-1	-1
1	2	-1
1	1	1

Rotate 180°

1	1	1
-1	2	1
-1	-1	1



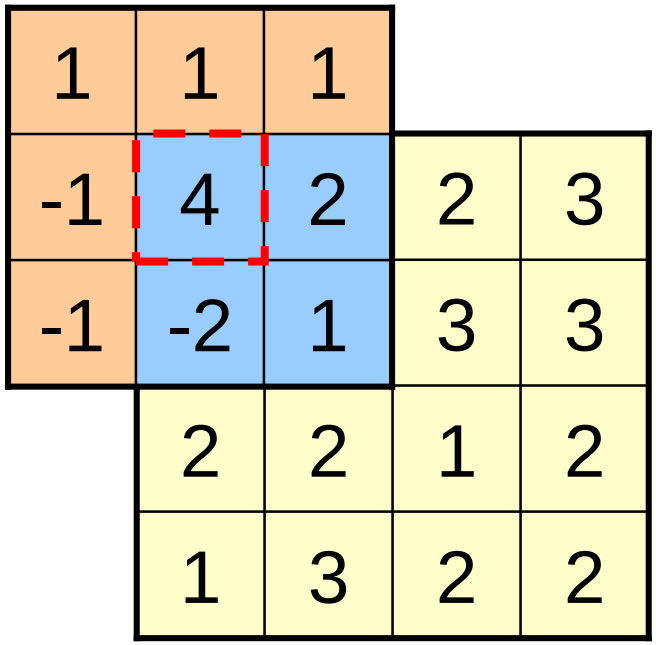
Input Image, f

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

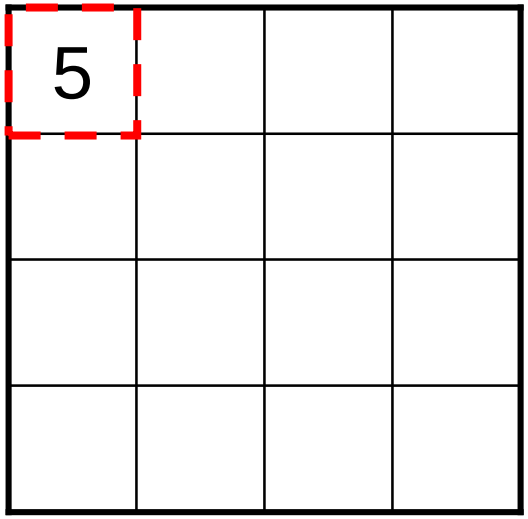
Convolution

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2



Input Image, *f*



Output Image, *g*

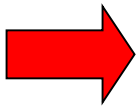
Convolution

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

1	1	1	
-2	4	2	3
-2	-1	3	3
2	2	1	2
1	3	2	2

Input Image, *f*



5	4		

Output Image, *g*

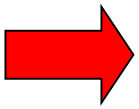
Convolution

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

	1	1	1
2	-2	4	3
2	-1	-3	3
2	2	1	2
1	3	2	2

Input Image, *f*



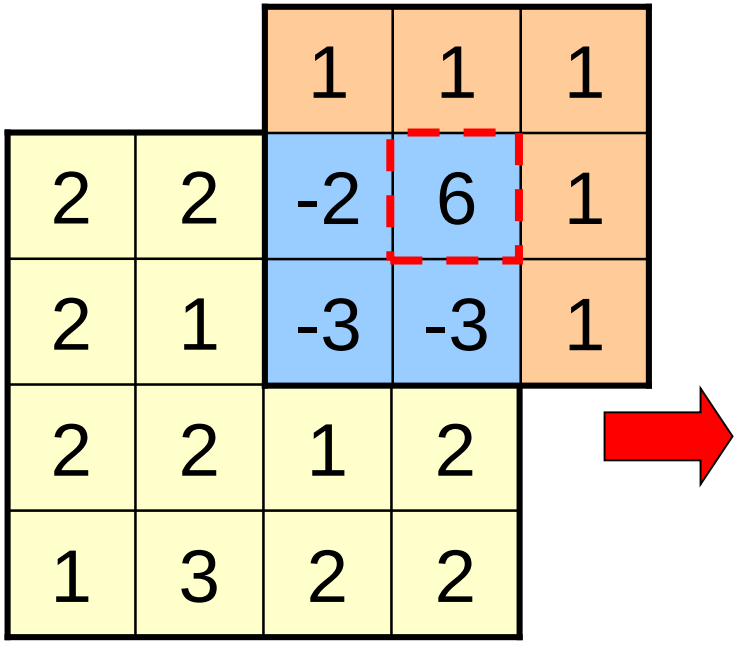
5	4	4	

Output Image, *g*

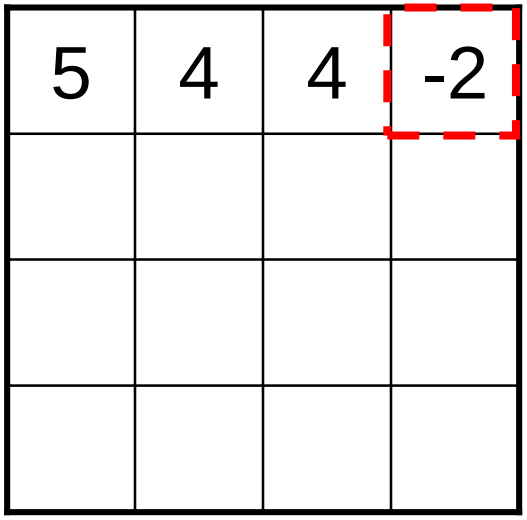
Convolution

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2



Input Image, *f*



Output Image, *g*

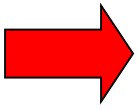
Convolution

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

1	2	2	2	3
-1	4	1	3	3
-1	-2	2	1	2
	1	3	2	2

Input Image, *f*



5	4	4	-2
9			

Output Image, *g*

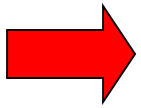
Convolution

1	1	1
-1	2	1
-1	-1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

2	2	2	3
-2	2	3	3
-2	-2	1	2
1	3	2	2

Input Image, *f*



5	4	4	-2
9	6		

Output Image, *g*

Convolution

5	4	4	-2
9	6	14	5
11	7	6	5
9	12	8	5

Final output Image, *g*

Correlation

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \omega(s,t) f(x+s, y+t)$$

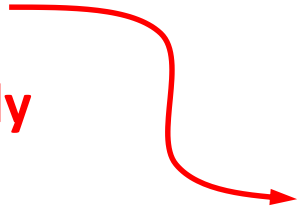
$$g = \omega \circ f$$

Correlation

correlation kernel, ω

1	-1	-1
1	2	-1
1	1	1

Don't rotate use it directly



Input Image f

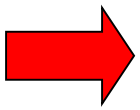
2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Correlation

1	-1	-1
1	2	-1
1	1	1

1	-1	-1		
1	4	-2	2	3
1	2	1	3	3
	2	2	1	2
	1	3	2	2

Input Image, f



5			

output Image, *g*

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

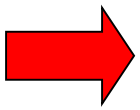
Correlation

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

1	-1	-1	
2	4	-2	3
2	1	3	3
2	2	1	2
1	3	2	2

Input Image, f



5	10		

output Image, *g*

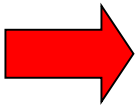
Correlation

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

	1	-1	-1
2	2	4	-3
2	1	3	3
2	2	1	2
1	3	2	2

Input Image, f



5	10	10	

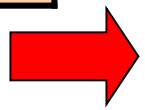
output Image, *g*

Correlation

1	-1	-1
1	2	-1
1	1	1

		1	-1	-1
2	2	2	6	-1
2	1	3	3	1
2	2	1	2	
1	3	2	2	

Input Image, f



5	10	10	15

output Image, *g*

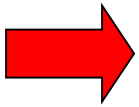
2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

Correlation

1	-1	-1
1	2	-1
1	1	1

1	-2	-2	2	3
1	4	-1	3	3
1	2	2	1	2
	1	3	2	2

Input Image, f



5	10	10	15
3			

output Image, *g*

2	2	2	3	
2	1	3	3	
2	2	1	2	
1	3	2	2	

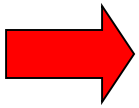
Correlation

1	-1	-1
1	2	-1
1	1	1

2	2	2	3
2	1	3	3
2	2	1	2
1	3	2	2

2	-2	-2	3
2	2	-3	3
2	2	1	2
1	3	2	2

Input Image, f



5	10	10	15
3	4		

output Image, *g*

Correlation

5	10	10	15
3	4	6	11
7	11	4	9
-5	4	4	5

Final output Image, *g*

Quiz # 4

For a 5x5 image array given here:

[[10, 10, 10, 10, 10],[10, 20, 20, 20, 10], [10, 20, 30, 20, 10], [10, 20, 20, 20, 10], [10, 10, 10, 10, 10]]

use a mask of 3x3 as below

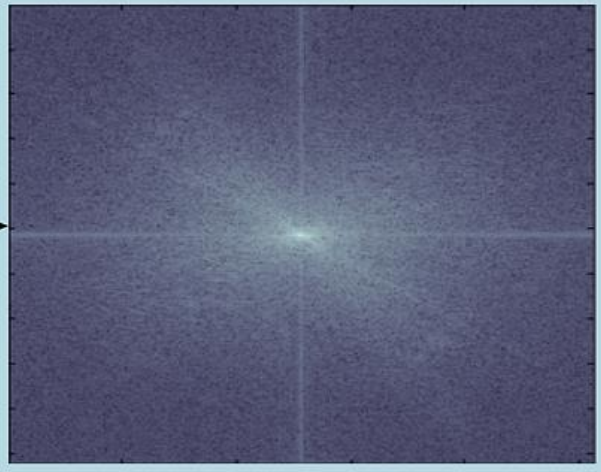
[[0, -1, 0], [-1, 5, -1], [0, -1, 0]]

to convolve and find output image array.

Quiz-4 Solution

[[40, 30, 40], [80, 70, 60], [20, 20, 30]]

Fourier Transform



Spatial Domain

Frequency Domain

Application of Averaging Filter

(a) Before processing

(b) After processing

Filtering as Convolution

- Process of linear filtering is similar to a frequency domain concept called convolution.
- For this reason, linear spatial filtering often is referred to as “convolving a mask with an image.”
- Similarly, filter masks are sometimes called convolution masks.
- The term convolution kernel also is in common use.

Smoothing Linear Filters

- The output (response) of a smoothing, linear spatial filter is simply the average of the pixels contained in the neighborhood of the filter mask. These filters sometimes are called averaging filters. They also are referred to a **low-pass filters.**
- By replacing the value of every pixel in an image by the average of the gray levels in the neighborhood defined by the filter mask, this process results in an image with reduced "sharp" transitions in gray levels.
- Because random noise typically consists of sharp transitions in gray levels, the most obvious application of smoothing is noise reduction.
- However, edges (which almost always are desirable features of an image) also are characterized by sharp transitions in gray levels, **so averaging filters have the undesirable side effect that they blur edges**

Smoothing Linear Filters

- A spatial averaging filter in which all coefficients are equal is sometimes called a *box filter.*

$$R = \frac{1}{9} \sum_{i=1}^{9} z_i.$$

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Smoothing Linear Filters

- The second mask yields a so-called *weighted average, terminology used to indicate that pixels are* multiplied by different coefficients, thus giving more importance (weight) to some pixels at the expense of others
- In the mask, the pixel at the center of the mask is multiplied by a higher value than any other, thus giving this pixel more importance in the calculation of the average

$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

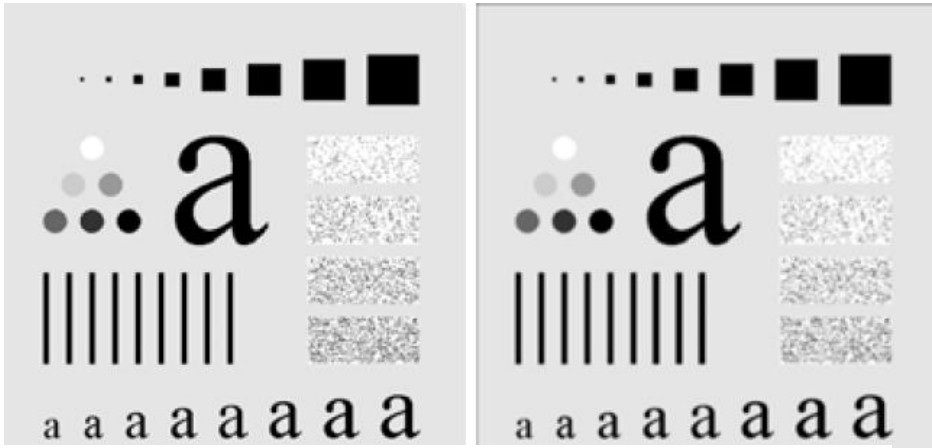
Smoothing Linear Filters

- In the mask, the pixel at the center of the mask is multiplied by a higher value than any other, thus giving this pixel more importance in the calculation of the average

$$g(x, y) = \frac{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t) f(x + s, y + t)}{\sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s, t)}$$

- The basic strategy behind weighing the center point the highest and then reducing the value of the coefficients as a function of increasing distance from the origin is simply an attempt to reduce blurring in the smoothing process

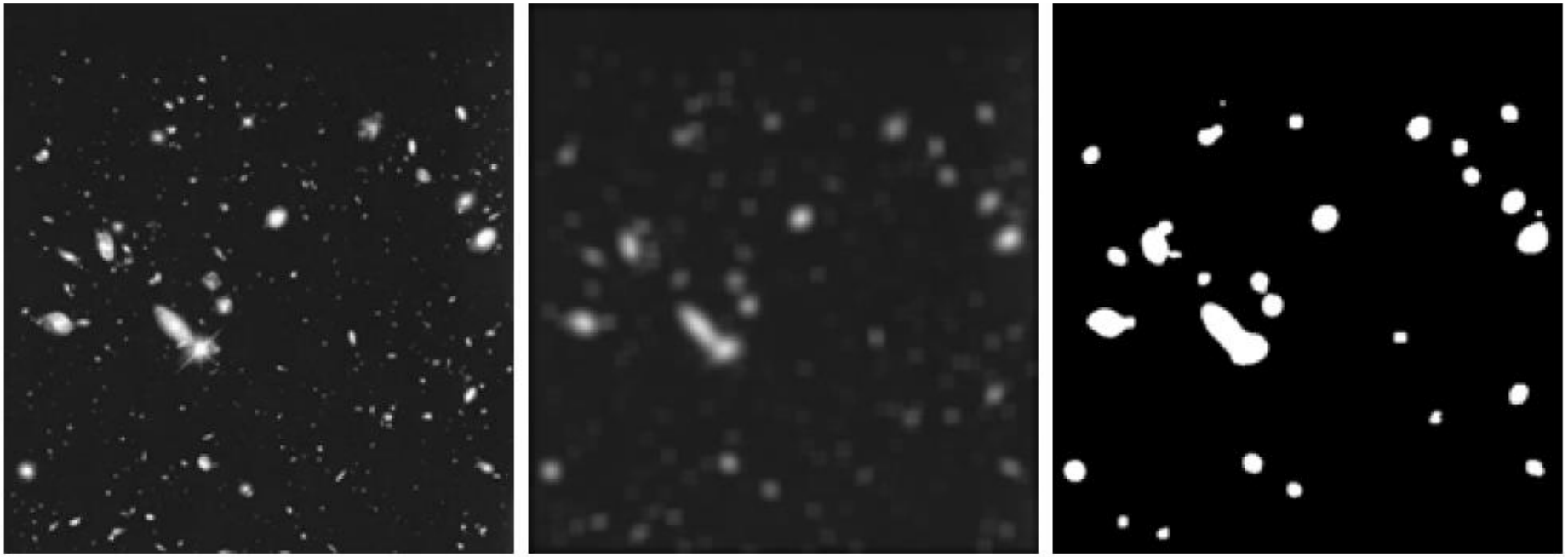
Smoothing Linear Filters



a b
c d
e f

FIGURE 3.35 (a) Original image, of size 500 × 500 pixels. (b)–(f) Results of smoothing with square averaging filter masks of sizes $n = 3, 5, 9, 15$, and 35, respectively. The black squares at the top are of sizes 3, 5, 9, 15, 25, 35, 45, and 55 pixels, respectively; their borders are 25 pixels apart. The letters at the bottom range in size from 10 to 24 points, in increments of 2 points; the large letter at the top is 60 points. The vertical bars are 5 pixels wide and 100 pixels high; their separation is 20 pixels. The diameter of the circles is 25 pixels, and their borders are 15 pixels apart; their gray levels range from 0% to 100% black in increments of 20%. The background of the image is 10% black. The noisy rectangles are of size 50 × 120 pixels.

Smoothing Linear Filters



a b c

FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

Low pass filtering (Averaging using 3x3 mask)

Low Pass Smoothing (using 5x5 mask)

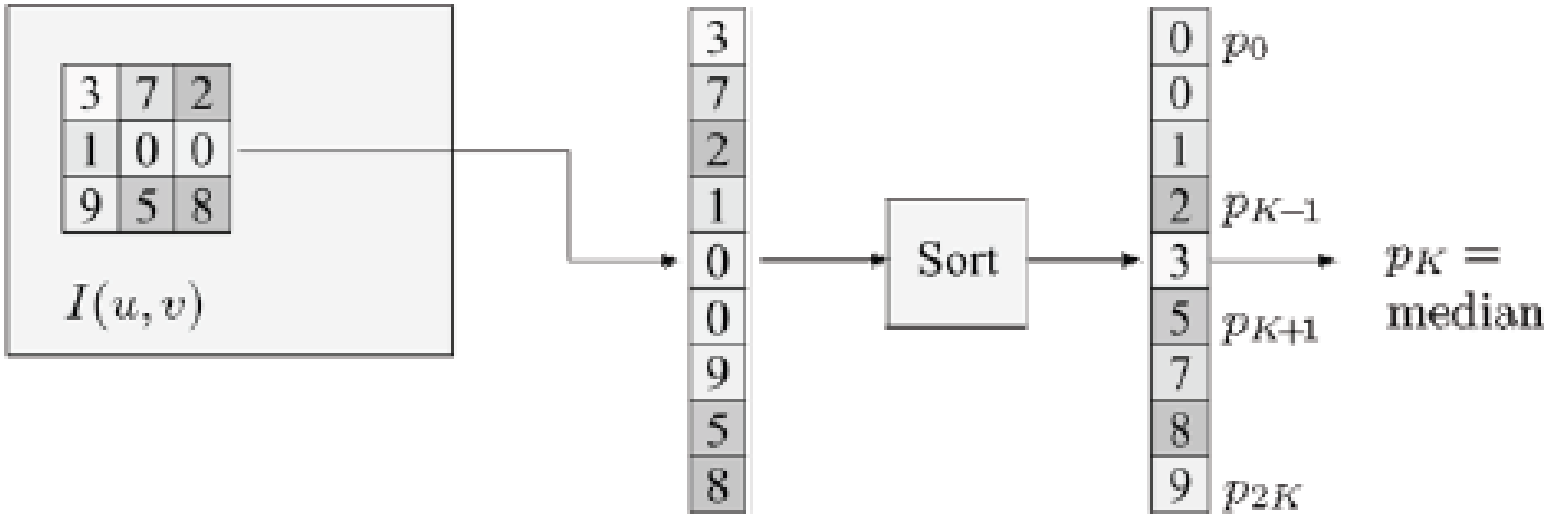
Order-statistics filters

- Order-statistics filters are nonlinear spatial filters whose response is based on ordering (ranking) the pixels contained in the image area encompassed by the filter, and then replacing the value of the center pixel with the value determined by the ranking result
- The best-known example in this category is the median filter, which, as its name implies, replaces the value of a pixel by the median of the gray levels in the neighborhood of that pixel (the original value of the pixel is included in the computation of the median)
- Median filters are quite popular because, for certain types of random noise, they provide excellent noise-reduction capabilities, with considerably less blurring than linear smoothing filters of similar size.
- Median filters are particularly effective in the presence of impulse noise, also called salt-and-pepper noise because of its appearance as white and black dots superimposed on an image

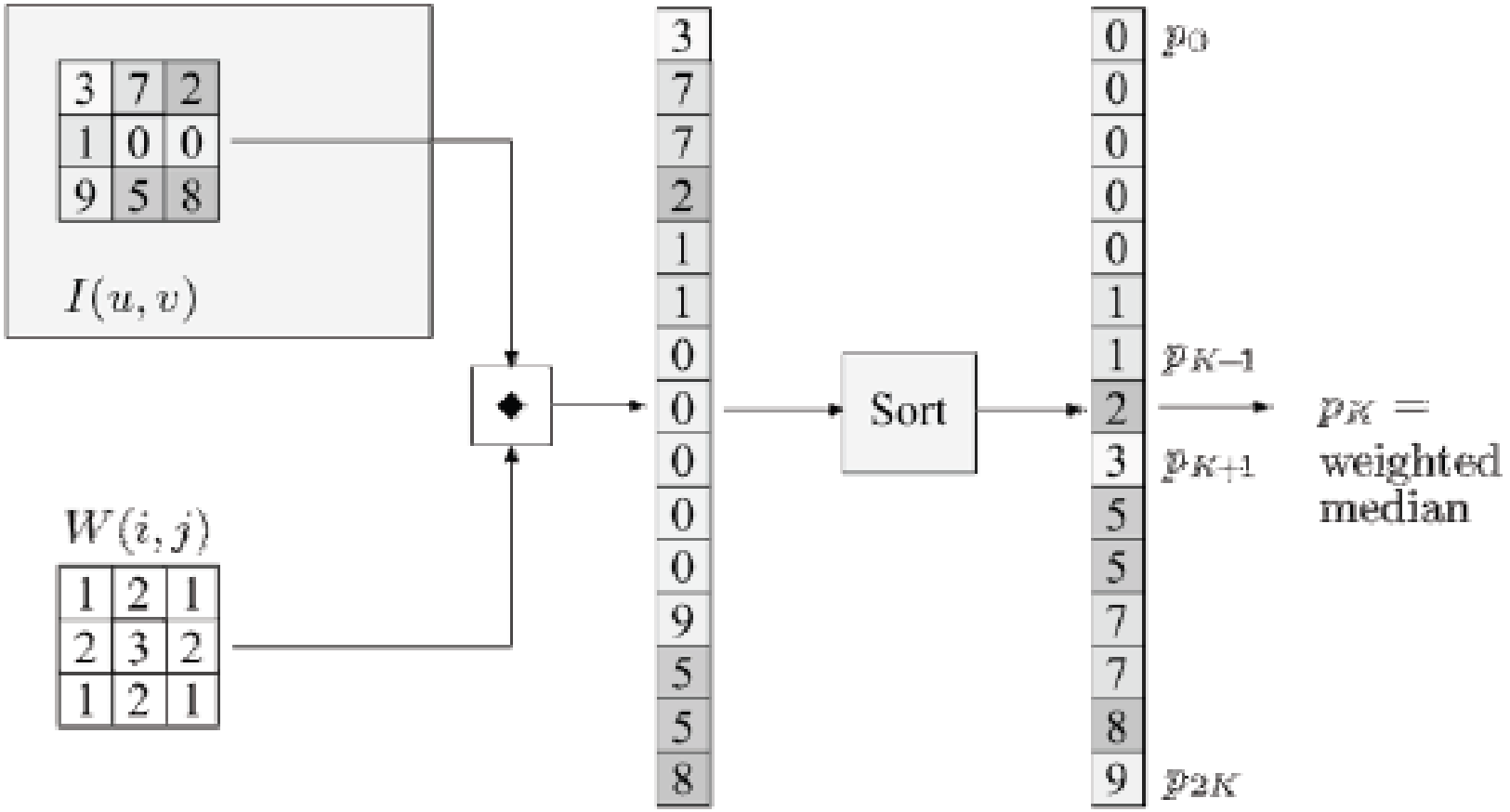
Median Filter

- The median of a set of values is such that half the values in the set are less than or equal to j, and half are greater than or equal to j. In order to perform median filtering at a point in an image, we first sort the values of the pixel in question and its neighbors, determine their median, and assign this value to that pixel.
- For example, in a 3x3 neighborhood the median is the 5th largest value, in a 5x5 neighborhood the 13th largest value, and so on.
- When several values in a neighborhood are the same, all equal values are grouped. For example, suppose that a 3x3 neighborhood has values (10, 20, 20, 20, 15, 20, 20, 25, 100).These values are sorted as (10, 15, 20, 20, 20, 20, 20, 25, 100), which results in a median of 20.
- **Thus, the principal function of median filters is to force points with distinct gray levels to be more like their neighbors.**

Median Filtering Algorithm

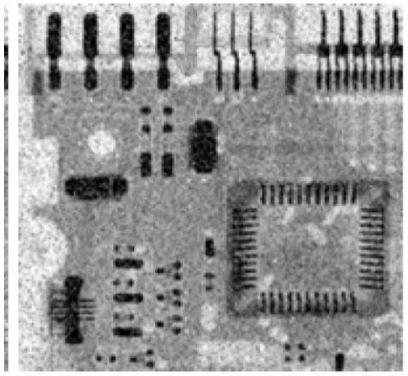


Weighted Median Filter

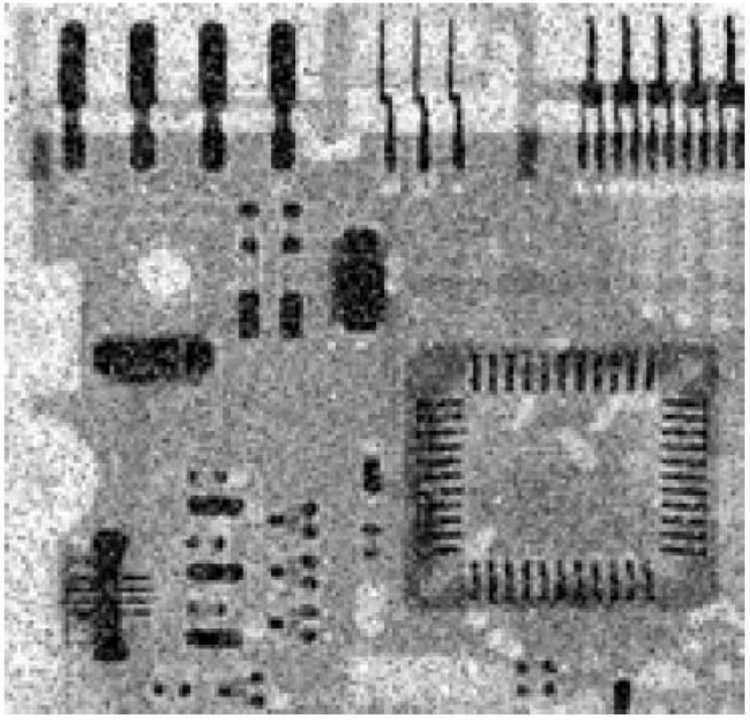


Comparison of Average versus Median Filter

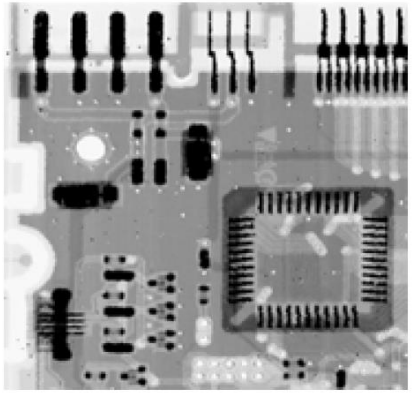
Use of averaging filter



Original (Source) Image



Use of median filter



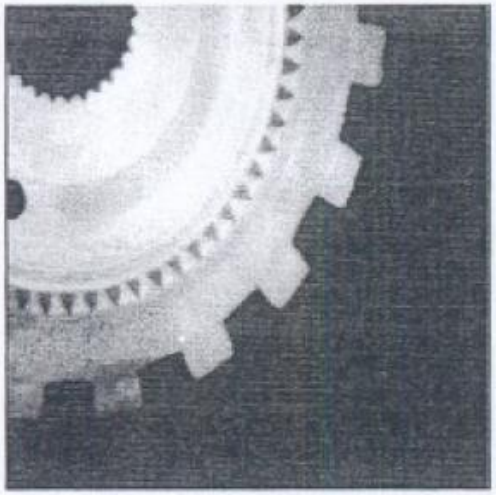
More on Noise & Filtering

Type of Image Noise

- Images are often corrupted by random variations in intensity values, called *noise.*
- *Some common types of noise are salt and pepper* noise, *impulse noise, and Gaussian noise.*
  - *Salt and pepper noise contains random* occurrences of both black and white intensity values.
  - Impulse noise contains only random occurrences of white intensity values.
  - Gaussian noise contains variations in intensity that are drawn from a Gaussian or normal distribution and is a very good model for many kinds of sensor noise,

Identify the Noise Type

- Original Images



- Noisy Images

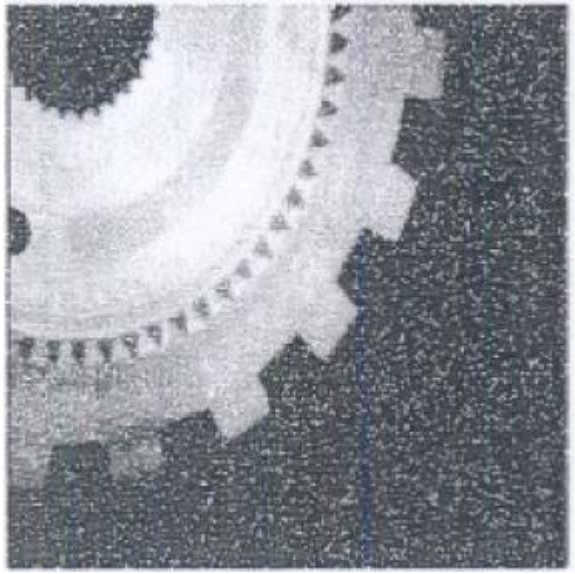


Image Noise Type-2

Original

Impulse noise

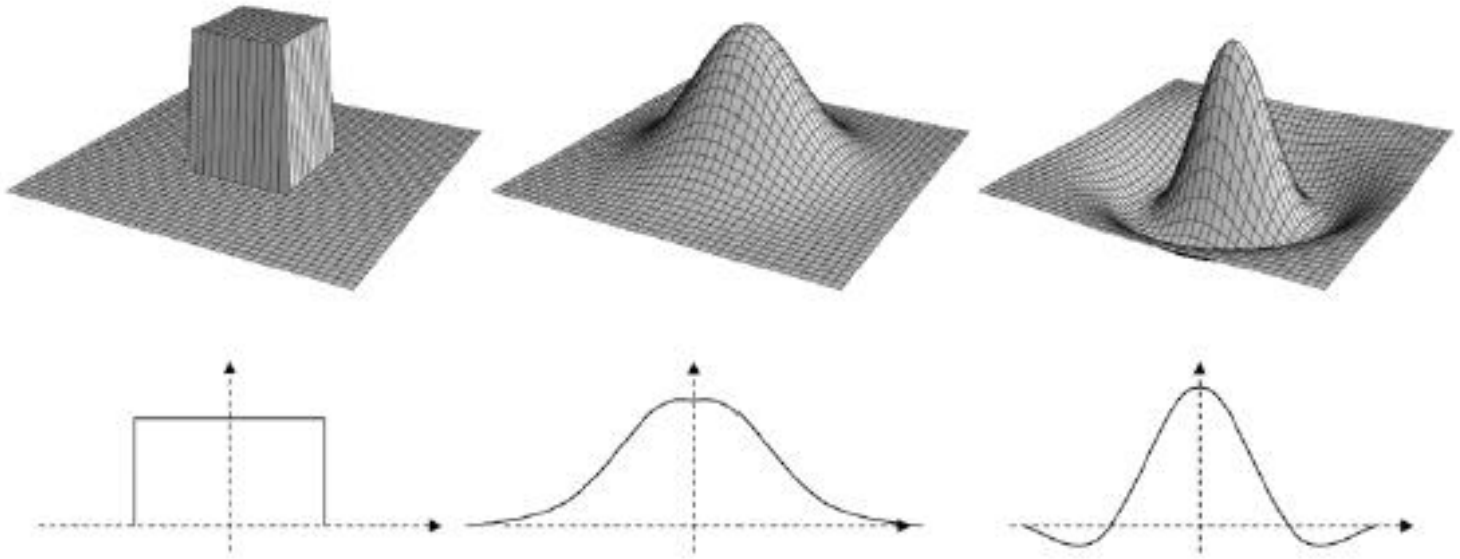
Gaussian noise

Salt and pepper noise

Linear versus Non-Linear Filters

- Linear smoothing filters are good filters for removing Gaussian noise and, in most cases, the other types of noise as well.
- A linear filter is implemented using the weighted sum of the pixels in successive windows.
- Typically, the same pattern of weights is used in each window, which means that the linear filter is spatially invariant and can be implemented using a convolution mask.
- If different filter weights are used for different parts of the image, but the filter is still implemented as a weighted sum, then the linear filter is spatially varying.
- Any filter that is not a weighted sum of pixels is a nonlinear filter.
- Nonlinear filters can be spatially invariant, meaning that the same calculation is performed regardless of the position in the image, or spatially varying. The median filter is a spatially invariant, nonlinear filter.

Types of Linear Filters



0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

(a)

0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0

(b)

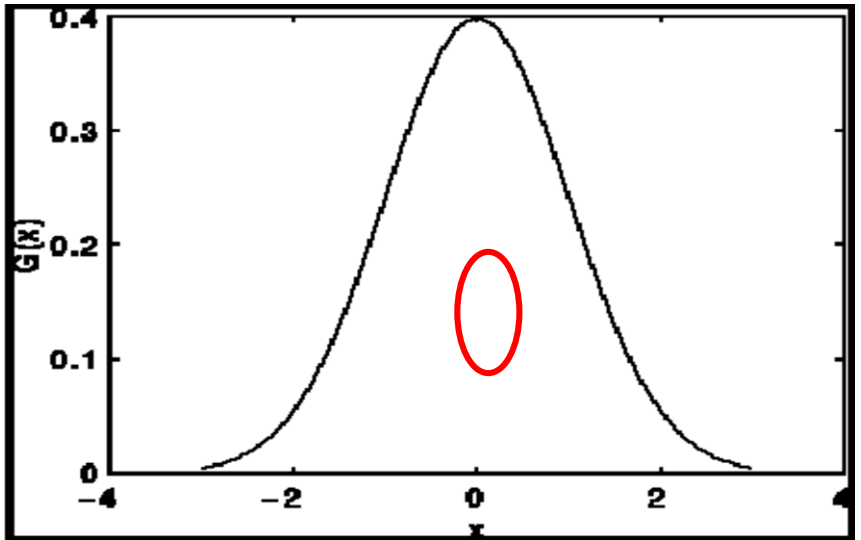
0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

(c)

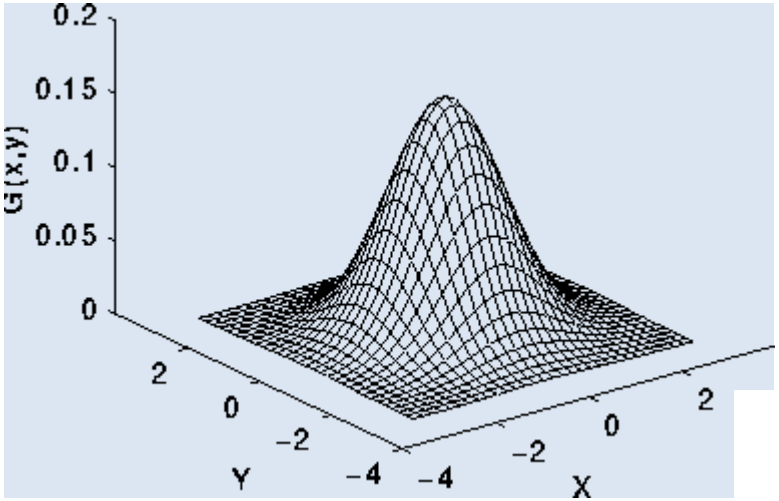
Gaussian Filters

- The Gaussian smoothing operator is a 2-D convolution operator that is used to `blur' images and remove detail and noise
- The Gaussian smoothing filter is a very good filter for removing noise drawn from a normal distribution

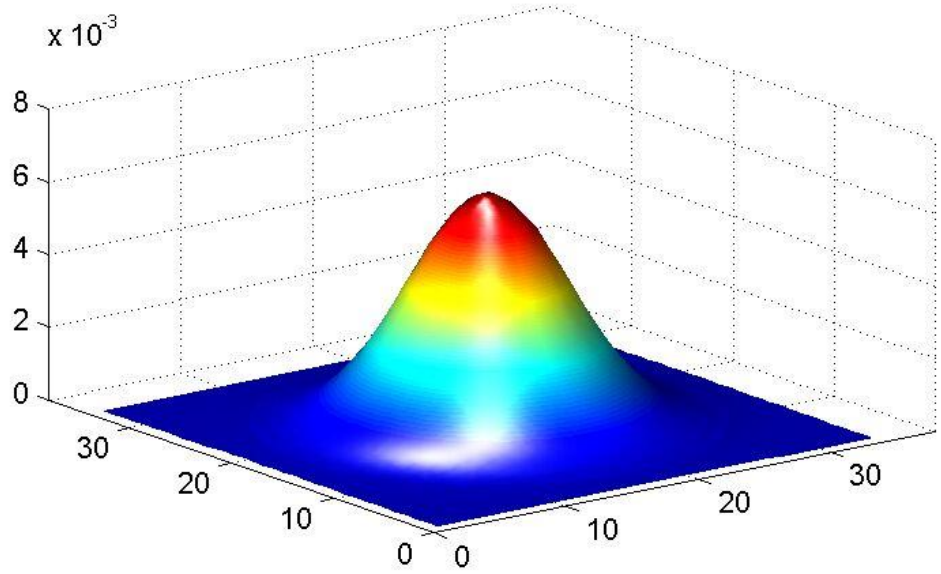
$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$$



2-D Gaussian



$$\left(G\,(x\,y) = \frac{1}{2\,\pi\,\sigma^2}\,e - \frac{x^2 + y^2}{2\,\sigma^2} \right)$$



Properties of Gaussian Filtering

- The degree of smoothing is determined by the standard deviation of the Gaussian.
- The Gaussian outputs a "weighted average" of each pixels neighborhood, with the average weighted more towards the value of the central pixels.
- This is in contrast to the mean filters uniformly weighted average. Because of this, a Gaussian provides gentler smoothing and preserves edges better than a similarly sized mean filter.
- Separable:

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2}) = (\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{x^2}{2\sigma^2})) * (\frac{1}{\sqrt{2\pi}\sigma}\exp(-\frac{y^2}{2\sigma^2}))$$

Convolving a Gaussian

$$G(x,y) * I(x,y) = G(x) * \{G(y) * I(x,y)\}$$

Designing Gaussian Filter

- Using Binomial Expansion which is an excellent approximation of Gaussian

$$(1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k$$

$$= \binom{n}{0} x^0 + \binom{n}{1} x^1 + \binom{n}{2} x^2 + \binom{n}{3} x^3 + \ldots\ldots\ldots + \binom{n}{n} x^n$$

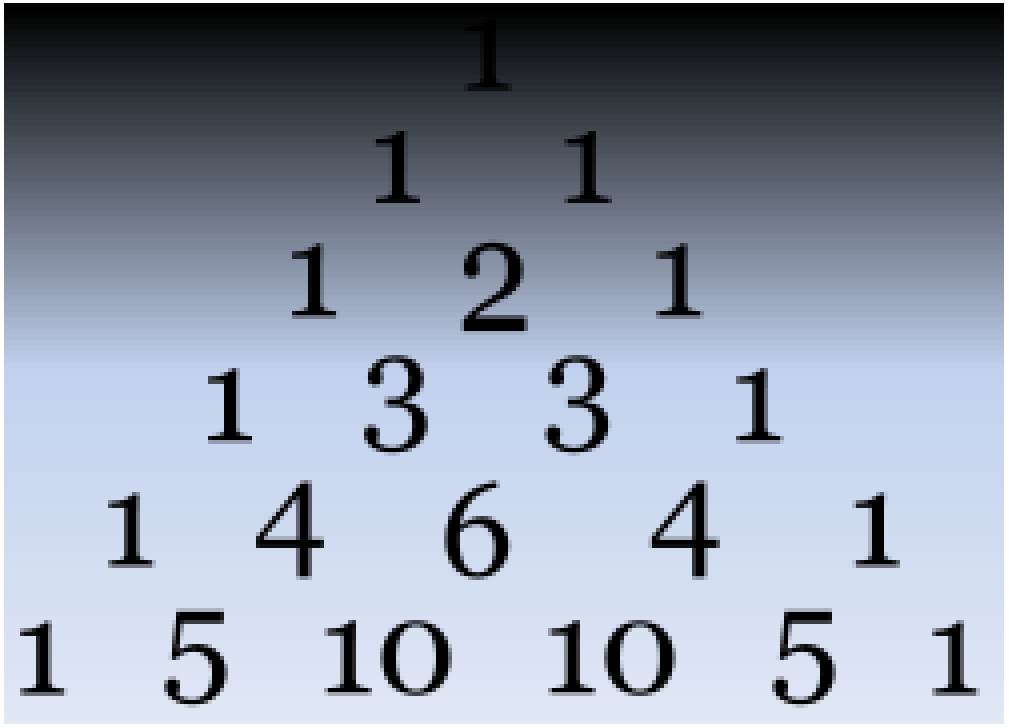
$$= 1 + \frac{n!}{1!(n-1)!} x + \frac{n!}{2!(n-2)!} x^2 + \ldots\ldots + \frac{n!}{n!(n-n)!} x^n$$

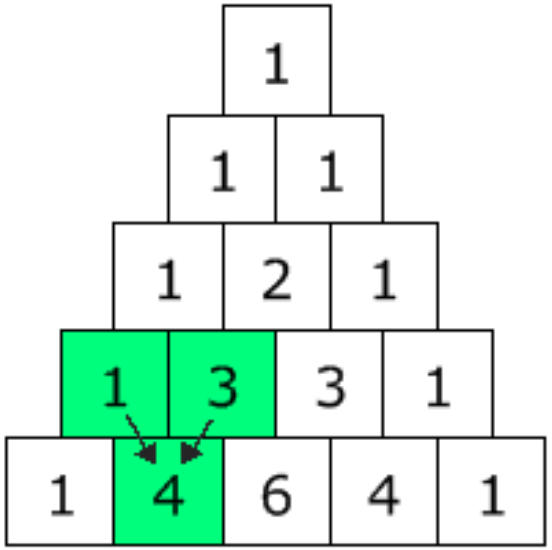
$$= 1 + nx + \frac{n(n-1)}{2} x^2 + \ldots\ldots + x^n$$

- Pascal Triangle:

1	4	6	4	1

Pascal Triangle





Designing Gaussian Filter

$$g[i,j] = ce^{-\frac{(i^2+j^2)}{2\sigma^2}}$$

$$\frac{g[i,j]}{c} = e^{-\frac{(i^2+j^2)}{2\sigma^2}}$$

Designing Gaussian Filter

- For a Gaussian mask of 7x7 with σ=2

$$\frac{g[i,j]}{c} = e^{-\frac{(i^2+j^2)}{2\sigma^2}}$$

$[i, j]$	-3	-2	-1	0	1	2	3
-3	.011	.039	.082	.105	.082	.039	.011
-2	.039	.135	.287	.368	.0287	.135	.039
-1	.082	.287	.606	.779	.606	.287	.082
0	.105	.368	.779	1.000	.779	.368	.105
1	.082	.287	.606	.779	.606	.287	.082
2	.039	.135	.287	.368	.287	.135	.039
3	.011	.039	.082	.105	.082	.039	.011

Designing Gaussian Filter

-

$[i, j]$	-3	-2	-1	0	1	2	3
-3	.011	.039	.082	.105	.082	.039	.011
-2	.039	.135	.287	.368	.0287	.135	.039
-1	.082	.287	.606	.779	.606	.287	.082
0	.105	.368	.779	1.000	.779	.368	.105
1	.082	.287	.606	.779	.606	.287	.082
2	.039	.135	.287	.368	.287	.135	.039
3	.011	.039	.082	.105	.082	.039	.011

$$\frac{g[3,3]}{k} = e^{-\frac{(3^2+3^2)}{2(2)^2}} = 0.011 \Rightarrow k = \frac{g[3,3]}{0.011} = \frac{1.0}{0.011} = 91$$

Designing Gaussian Filter

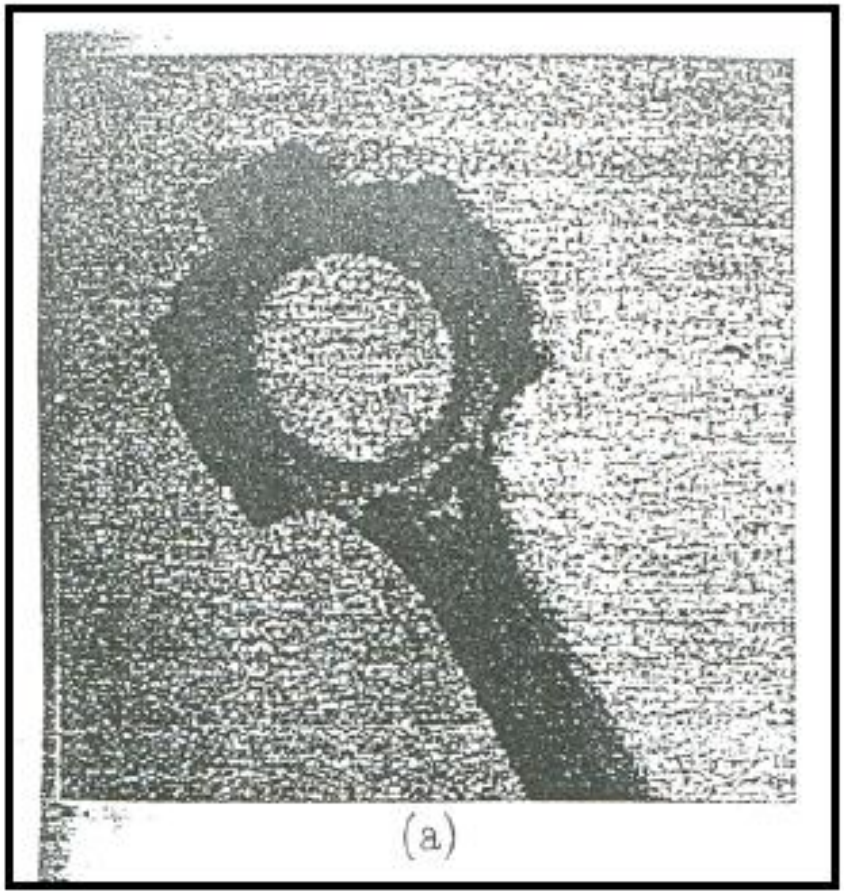
- For a Gaussian mask of 7x7 with σ=2

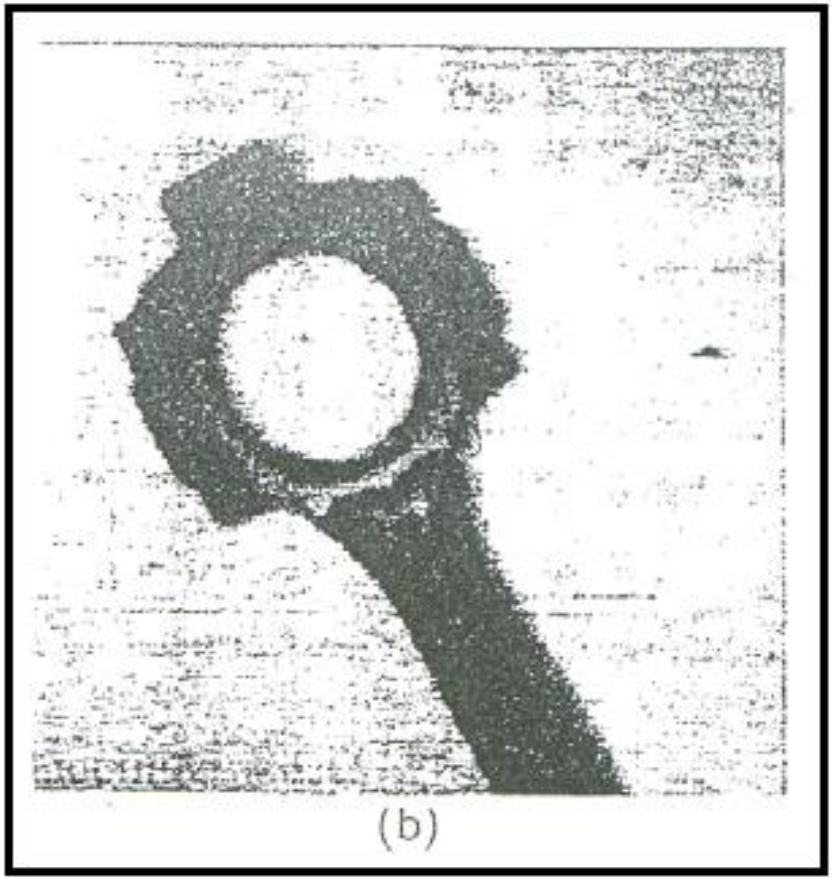
$[i, j]$	-3	-2	-1	0	1	2	3
-3	.011	.039	.082	.105	.082	.039	.011
-2	.039	.135	.287	.368	.0287	.135	.039
-1	.082	.287	.606	.779	.606	.287	.082
0	.105	.368	.779	1.000	.779	.368	.105
1	.082	.287	.606	.779	.606	.287	.082
2	.039	.135	.287	.368	.287	.135	.039
3	.011	.039	.082	.105	.082	.039	.011

$[i, j]$	-3	-2	-1	0	1	2	3
-3	1	4	7	10	7	4	1
-2	4	12	26	33	26	12	4
-1	7	26	55	71	55	26	7
0	10	33	71	91	71	33	10
1	7	26	55	71	55	26	7
2	4	12	26	33	26	12	4
3	1	4	7	10	7	4	1

$$\frac{g[3,3]}{k} = e^{-\frac{(3^2+3^2)}{2(2)^2}} = 0.011 \Rightarrow k = \frac{g[3,3]}{0.011} = \frac{1.0}{0.011} = 91$$

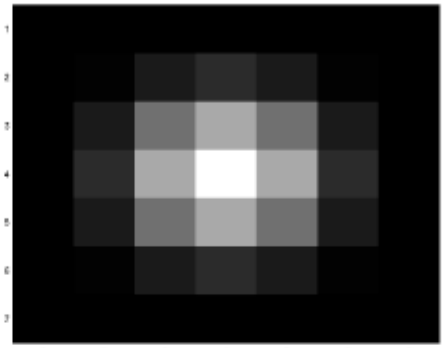
Gaussian Smoothed Image





Gaussian Filtering in MATLAB

```
>> sigma = 1

sigma =

     1

>> halfwid = 3*sigma

halfwid =
```

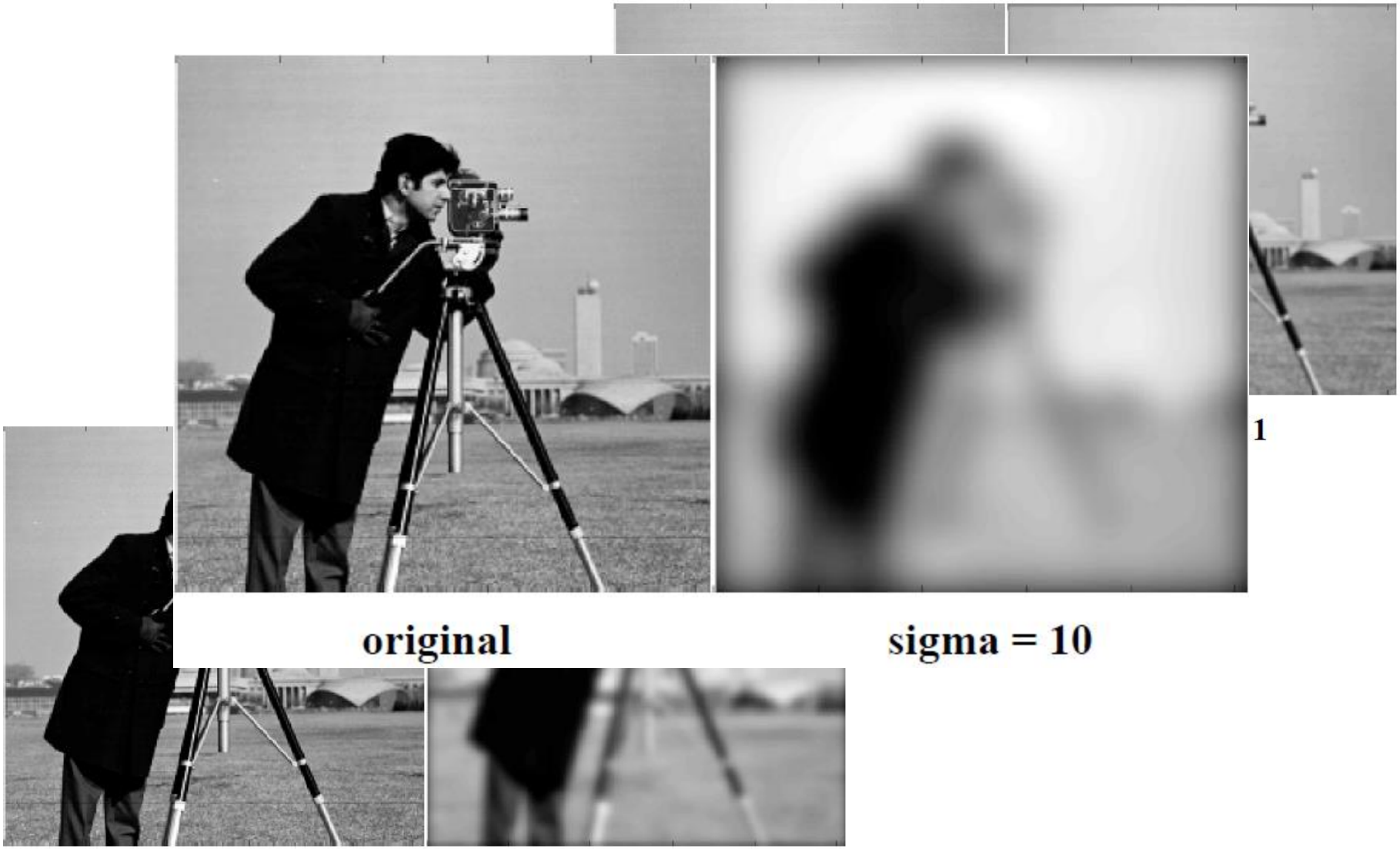


```
>> sigma = 1

sigma =

     1

>> halfwid = 3*sigma

halfwid =

     3
>> [xx,yy] = meshgrid(-halfwid:halfwid, -halfwid:halfwid);
>> gau = exp(-1/(2*sigma^2) * (xx.^2 + yy.^2))

gau =

    0.0001    0.0015    0.0067    0.0111    0.0067    0.0015    0.0001
    0.0015    0.0183    0.0821    0.1353    0.0821    0.0183    0.0015
    0.0067    0.0821    0.3679    0.6065    0.3679    0.0821    0.0067
    0.0111    0.1353    0.6065    1.0000    0.6065    0.1353    0.0111
    0.0067    0.0821    0.3679    0.6065    0.3679    0.0821    0.0067
    0.0015    0.0183    0.0821    0.1353    0.0821    0.0183    0.0015
    0.0001    0.0015    0.0067    0.0111    0.0067    0.0015    0.0001
```

```
-halfwid:halfwid);
.^2))
```

```
    0.0067    0.0015    0.0001
    0.0821    0.0183    0.0015
    0.3679    0.0821    0.0067
    0.6065    0.1353    0.0111
    0.3679    0.0821    0.0067
    0.0821    0.0183    0.0015
    0.0067    0.0015    0.0001
```

Gaussian Smoothing Levels



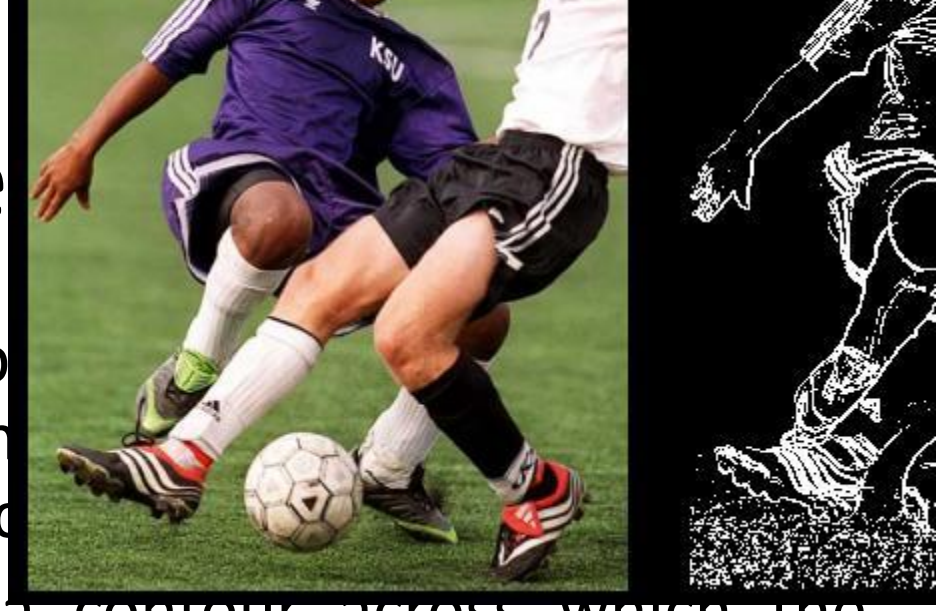
Gaussian Smoothing Levels

- The std. dev σ of the Gaussian determines the amount of smoothing.

Low Pass versus High Pass Filtering

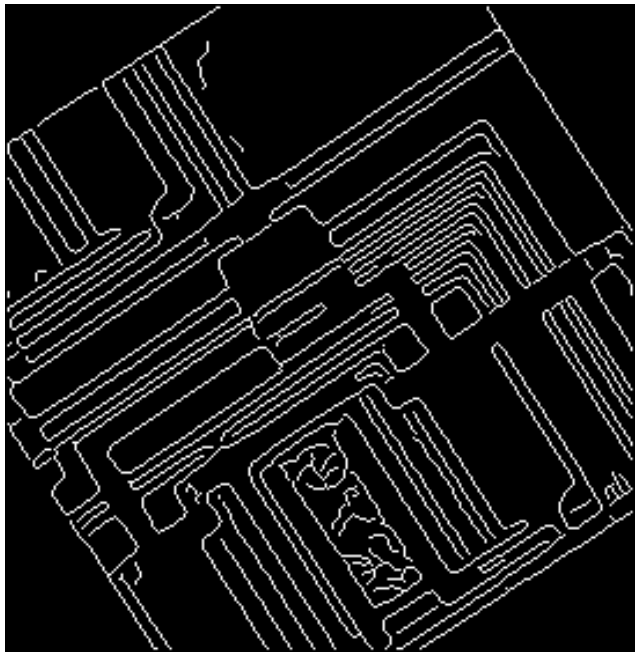
- Low pass filtering is used for smoothing, in a particular elimination of image noise.
- High pass filtering is used for contour enhancement

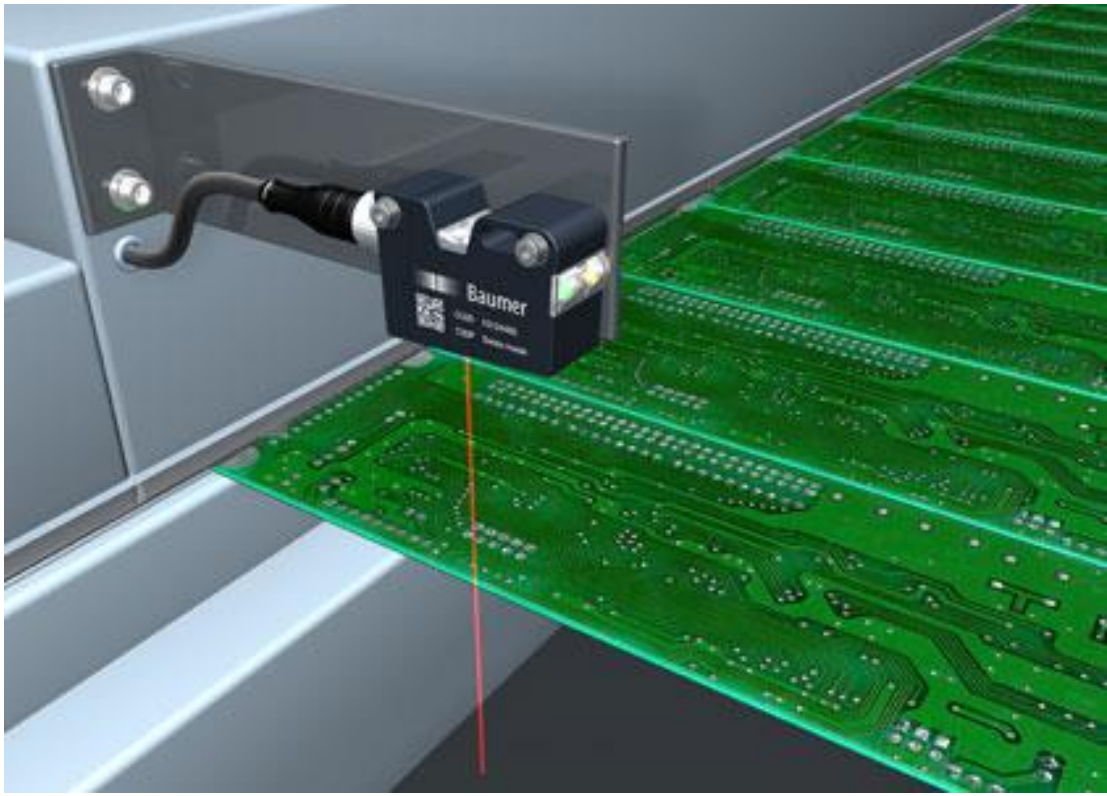
Edge & Line



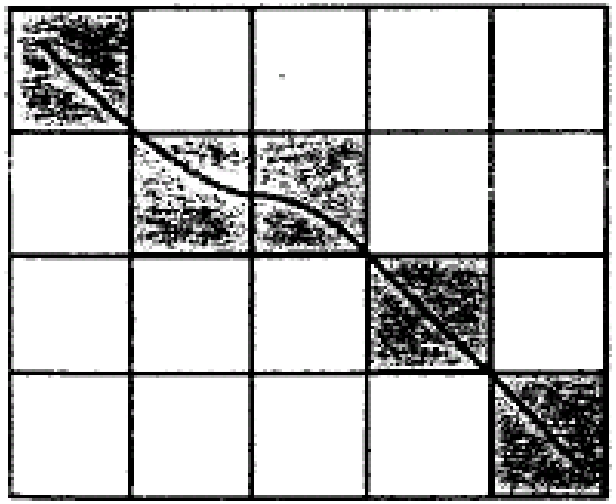
- Edge detection is an imp
  number of image processin
  segmentation, character rec
- An edge in an image is a contour across which the brightness of the image changes abruptly in magnitude or in the rate of change of magnitude.
- An *edge is* characterized by an abrupt change in intensity indicating the boundary between two regions in an image, and is often referred to as a discontinuity
- A ***line*** *is* a region of constant intensity found between two edges which act as a boundary for the line

Edge & Line Detection

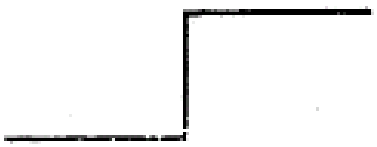




Edges & Lines



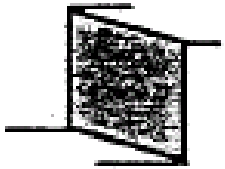
Connected edge points



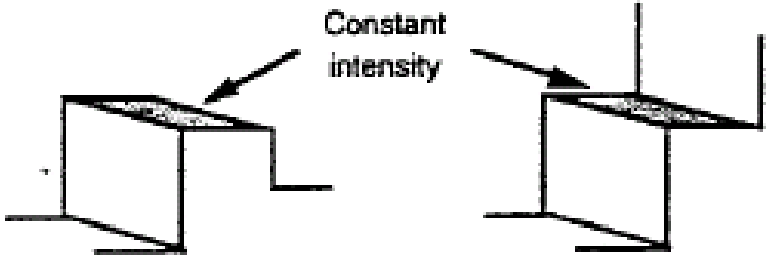
Ideal edge

Real edge

Noisy edge



Edge



Lines – a pair of edges which act as the boundary of a thin region of constant intensity

Edge Enhancement

- Another class of filters can be defined by their effect of emphasizing or strengthening the edges within an image.
- In general a high pass filter has the inverse characteristic of a low pass filter; it will not change the high frequency component of the signal but will attenuate the low frequencies and eliminate any constant background intensity.
- High pass filter has the property that pixel data in the region of an object edge is modified and the edge effect is enhanced.
- This property of edge detection is a prerequisite for many image analysis operations.